Where am I from? –East Asian Ethnicity Classification from Facial Recognition

Haoxuan Chen*[1], Yiran Deng†[1] and Shuying Zhang‡[1]

[1]Stanford University

Abstract

We implemented the machine learning approach to classify and predict Chinese, Japanese and Korean based on the facial images. A series of computational approaches were used to train the classifier, including a k-nearest Neighbor algorithm, a Support Vector Machine, a two-layer neural network and a convolutional neural network (CNN). Of all the methods we proposed, CNN had the highest prediction accuracy(89.2%) in the 3-class classification.

Keywords— Ethnicity Classification among East Asians, Facial Recognition, Computer Vision, Machine Learning

1 Introduction and Related Work

Face conveys the most direct and fastest impression of an individual. We can often guess a person's ethnicity by the way he or she looks. However, Chinese, Japanese and Korean have very similar facial anatomy partially due to their close geographical relationships. Often times people from these three countries encounter situations where others mistaken their nationality the first time they meet. This is very common in the United States, where people from all parts of the world mingle and connect with each other, and differentiating people from these three countries purely based on facial features can be particularly difficult. Interestingly, some people claim that they can differentiate these three subgroups of east Asians by the way they look. It raises the question that if facial traits per se give sufficient intuition on the nationality of an east Asian. Fu et. al [1] have previously demonstrated the possibility of using machine learning methodology to classify ethnicity utilizing facial features among ethnic groups with more discerning facial characteristics (such as comparing Asian, Caucasian and Africans).

Gleaned the power of the recent advances in computer vision and machine learning, We took the challenge to investigate whether or not it is possible to classify East Asians whose faces are highly resembling and whether or not there might be a mathematical rationale behind these nuances that are not so obvious to human eyes.

*haoxuan@stanford.edu
†yrdeng@stanford.edu
‡shuyingz@stanford.edu

2 Methods

2.1 Classification Pipeline

The goal of the study is to predict if a person is a Chinese, Japanese or Korean given his or her facial image. In order to make such predictions, we collected a set of facial images categorized into Chinese, Japanese and Korean with labeled gender. We randomly selected 80% as the training data to train the machine learning algorithm, 10% as validation set and 10% as test set. The facial images were first cropped and augmented and fed into different learning approaches. The training data are labeled in either 3 classes (Chinese, Japanese, Korean)or 6 classes (Chinese, Japanese, Korean with gender discrimination). The details of data pre-processing can be found in the section Dataset and Preprocessing.

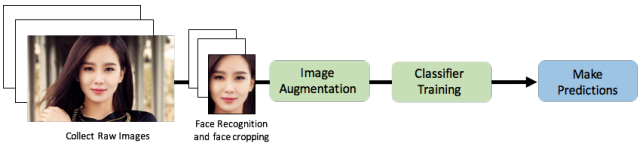

Figure 1: classification pipeline.

2.2 k-Nearest Neighbor(kNN) Classifier

We first attempted a k-Nearest Neighbor (kNN) classifier. The algorithm is executed as follows:

Input : the training data $D \in \mathbb{R}^{m \times n}$, where m is the number of images, n is the number of pixels per image; the test data $X \in \mathbb{R}^{m \times n}$; L, the set of classes used to label the objects.

Output : $y \in L$ gives the label of the ith test data x^i.

For each : $x^i \in X$ and $d^j \in D$, **do**:

Calculate the L2 distance between ith test point x^i and the jth training point y^j and store the result in $dist[i, j]$.

Given all the distances between test points and training points in the matrix $dist$,

For each : x^i, **do**:

Find the k nearest neighbors of the ith training point x^i, find the labels of these neighbors, find the most common label in the list of labels. Assign this label to the

test data. If more than one label gets the same amount of votes, break ties by choosing the smaller label.

2.3 Support Vector Machine

The support Vector Machine approach is a classical supervised learning method used to separate data points in the high-dimensional space. Given a set of datapoints $x^{(i)} \in \mathbb{R}^n$ with labels $y^{(i)} \in \{0, 2\}$ or $y^{(i)} \in \{0, 5\}$, the hinge loss is:

$$\frac{1}{N} \sum_i \sum_{j \neq y_i} [\max(0, w_j^T x_i - w_{y_i}^T x_i + \Delta)] + \lambda \|w\|_2^2 \quad (1)$$

The update rule follows stochastic gradient descent. We also conducted a hold-out cross-validation to attenuate potential over-fitting.

2.4 Two-Layer Neural Network

We proposed two neural network approaches to predict nationality. First we constructed a two-layer neural network. The network architecture is shown in figure 2. Each pre-processed facial image is flatten to a row vector which has a dimension of $1 \times (128 \cdot 128 \cdot 3)$ (RGB) or $1 \times (128 \cdot 128)$ (grayscale), then all images are stacked together and fed into the network. The network has a Rectified Linear Unit (ReLU) layer sandwiched by two fully connected layers. The ReLU layer computes the function:

$$f(x) = \max(0, x) \quad (2)$$

The hidden layer dimension (number of neurons) is tuned by cross validation and chosen to be 64. After the second fully connected layer, the output label is predicted by a softmax classifier.

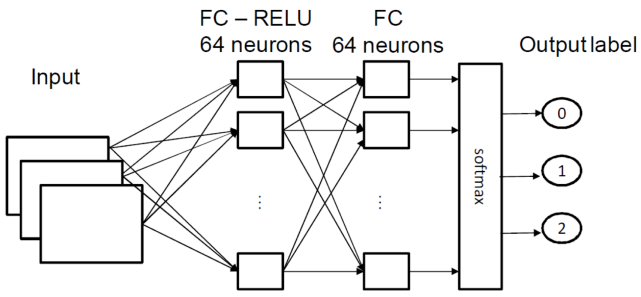

Figure 2: Architecture of the 2-layer neural network

.

2.5 Convolutional Neural Network

Next, we built and trained a convolutional neural network (CNN) using TensorFlow Library[2]. We built a 5-layer CNN whose architecture is shown in Figure 3. The CNN consists of two convolutional layers (with 64 and 128 filters respectively), one fully-connected layer (with 1024 neurons) and a dropout layer, followed by an output layer and we applied L2 regularization to each layer. Since we didn't manage to get a GPU working for our model, we resized the image to 64×64 for training. By tuning the hyperparameters such as learning rate, dropout rate and regularzation strength, we tried to improve the accuracy of our CNN.

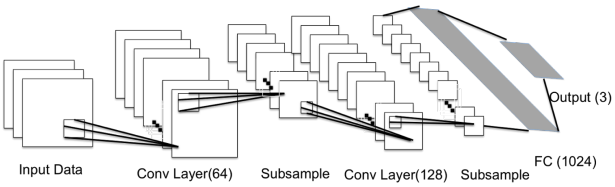

Figure 3: Architecture of the convolutional neural network

.

3 Dataset and Preprocessing

3.1 Face Attribute Dataset Collection

Despite that there is a wealth of large-scare facial image databases available, such as Chicago Face Database and the CAS PEAL database, nevertheless, these databases are not applicable to our objective, as the images from these datasets are labeled by ethnicity instead of nationality. Therefore, we manually harvested a total of 1380 profile photos of university faculty members and public figures from the three countries. All images were directly downloaded from the internet. We collected 270 Japanese facial images from the Univeristy of Tokyo [3] and Waseda University[4]; 260 Korean facial images from Seoul National University[5]; 249 Chinese facial images from Shanghai Jiaotong University[6] and Sun Yat-Sen University[7]. Next, we utilized the Google Custom Search API[8] to collect profile photos of celebrities grouped by nationality using full names as the search keyword. The number of facial images for each subgroup is listed in Table 1.

3.2 Image Preprocessing

Image preprocessing was conducted before training a classifier (Figure 4).The input facial images can be of varying sizes. We need to normalize the faces so that each training datapoint would have the same dimension. We first cropped the face using Haar Cascade Face Detection algorithm packaged with OpenCV[9] such that the noise from the image background is eliminated. And then we reframed the faces to 64×64 images for the CNN and 128×128 images for the rest of the methods. This is because a larger image significantly bogged down the computational speed for neural network approaches. Afterwards we applied different preprocessing methods, including converting to grayscale and mean substraction.

Converting images to grayscale is a powerful way of eliminating the differences of brightness and contrast in the input. We converted the input images to grayscale, which reduces the input dimension by 3 times.

Mean subtraction is the most common form of preprocessing. It involves subtracting the mean across ev-

Table 1: **Number of images from each country by gender**

Gender	Chinese	Japanese	Korean
Male	209	314	280
Female	209	184	186

ery individual feature in the data, and has the geometric interpretation of centering the cloud of data around the origin along every dimension. With images specifically, for convenience it can be common to subtract a single value from all pixels, or to do so separately across the three color channels[10].

Since the size of our dataset is not big engough, we apply data augmentation by flipping each image and added and subtracted a randomnized brightness value to the image. As a result we quardrupled our dataset before feeding them into the classifiers.

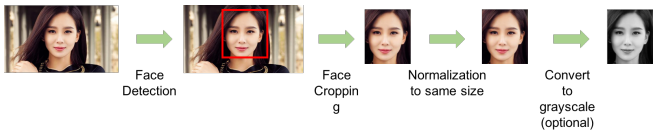

Figure 4: The workflow of image preprocessing.

4 Results and Discussion

We used all the aforementioned methods to predict the labels of our test set, the test accuracy is listed in Table 2. For cropped images divided into 3 classes, CNN outstrips other classifiers by a large margin (a 89.2% accuracy). While kNN, SVM, and 2-layer neural network have similar results. For 6 classes, the two types of neural network all have a test accuracy much higher than the baseline (74.4% and 83.5%, baseline is 16.7%). If comparing the test accuracy to baseline, we observe that 6 classes have a better performance. This is probably because male and female have distinct facial features, such as hair, facial form and skin color, thus it is better to seperate these two classes.

For two-layer neural network, we used several preprocessing methods to compare their effects on uncropped and cropped images. We first used original inputs as benchmark, then we converted all the images to grayscale. For mean subtraction, we computed a mean face over the training examples, then we subtracted it from all the images in the training, validation and test set. When we applied both methods, we first converted all the images to grayscale then repeated the process of mean substraction. The results are shown in Table 3 and Figure 5.

By tuning the hyperparameters, we found that a learning rate of 1.25×10^{-4}, a regularization parameter of 0.25, an iteration of 6500, and a batch size of 50 yield the best result.

The figure indicates that cropping is an effective way of classify facial images as all the accuracies of different pre-processing methods increase after cropping. Using only mean subtraction yields the worst prediction accuracy. While using both mean subtraction and grayscale images yield decent results for both uncropped and cropped images, indicating that the reduced feature dimension can limit overfitting and lead to a higher accuracy. The highest accuracy we can get using two-layer neural network is 74.4%.

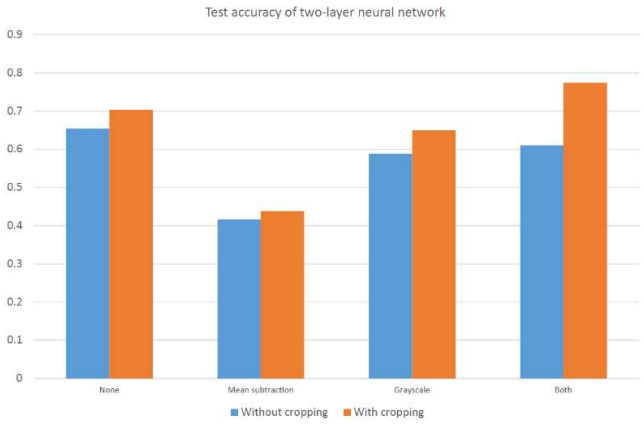

Figure 5: The prediction results for 2-layer neural nework with different pre-processing methods.

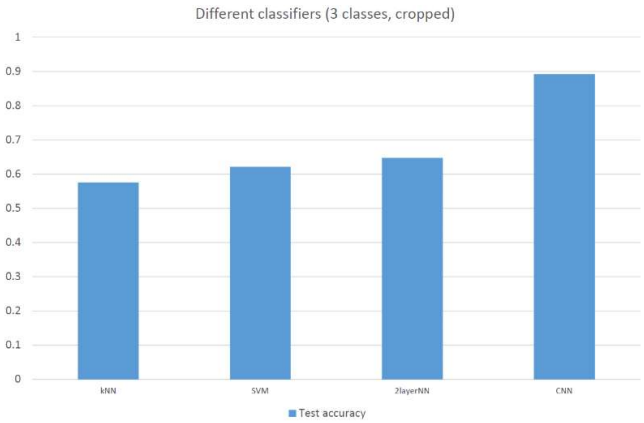

Figure 6: The 3-class prediction accuracy of the test data using different approaches.

The Convolutional Neural Network acheived a high testing accuracy of 89.2% for our original dataset. To see if it is also applicable to classify other faces CNN has not seen before, we collected 246 photos from our friends and choosing images randomly from Google and used CNN to make predictions. We then got an accuracy of 61.3% and the confusion matrix is shown in Figure 8. We can see the CNN model is overfitting a lot. That is because the number of the features of our images is $64 \times 64 \times 3 = 12288$, but the size of our training set is only 5520, which is much smaller than the feaures size. Moreover, the overfitting problem may be

Table 2: **Comparison between proposed approaches**

Number of classes[1]	kNN	SVM	2-Layer	CNN
3	57.5%	62.1%	64.7%	89.2%
6	50.9%	48.2%	74.4%	83.5%

Table 3: **Comparison between different pre-processing methods in two-layer neural network**

Test accuracy / Gender accuracy	None	Mean subtraction	Grayscale	Both
Without cropping	65.4%/88.6%	41.6%/75.3%	58.8%/75.3%	61.0%/86.0%
With cropping	70.3%/93.1%	43.8%/81.4%	64.9%/93.5%	74.4%/96.0%

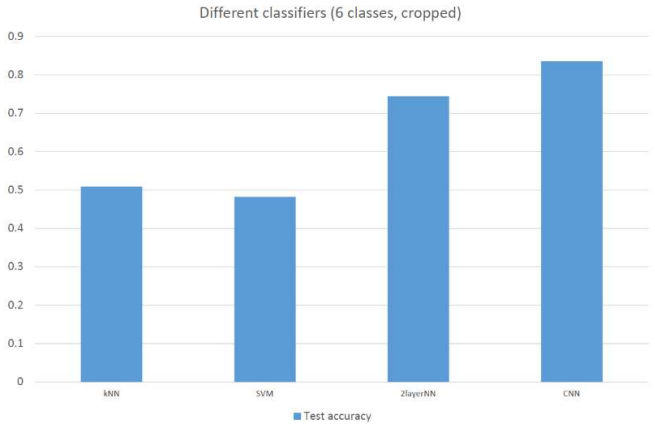

Figure 7: The 6-class prediction accuracy of the test data using different approaches.

attributed to the fact that our images are not representative enough since most of the images are from celebrities, who ususally have makeup on their faces and are better looking. Therefore the model is prone to overfitting. The non-availablity of GPU is another obstacle for the training process, giving us limited space to tune the parameters and the architecture.

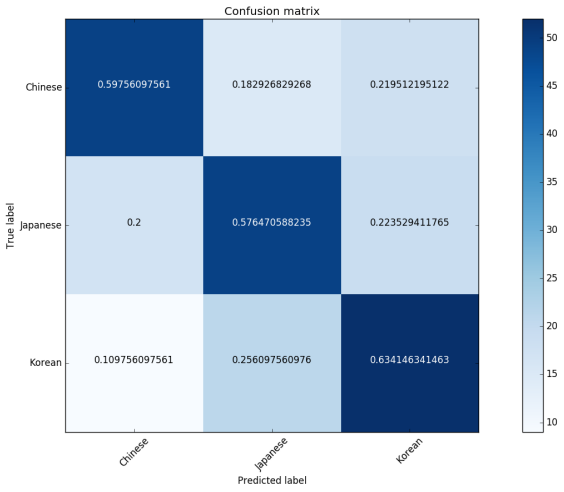

Figure 8: Confusion matrix of CNN.

5 Other Things We've Tried

5.1 Unsupervised Learning

We also attempted one unsupervised learning approach, namely the k-means clustering. Since this is a classification problem, it can be treated as a clustering problem. We picked the number of clusters k to be 3. However, our algorithm failed to cluster the classes correctly due to background noises and similarities between these 3 classes.

5.2 Principle Component Analysis

We attempted using the principle component analysis (PCA). The idea behind PCA is that by mapping the data points to a lower dimensional space we can categorize data into a set number of classes. Literature has shown that PCA can reliably categorize facial images by race via reducing the dimensions of facial cues. Phillips and OToole et al. have investigated PCA-based methods to discriminate between Japanese and Caucasian facial images, and the results seemed relatively promising (80% accuracy)[11].

We also attempted PCA, however, the step that computes the covariance matrix of the data is too slow such that it encumbers the advantage of reduced dimension. Therefore, we gave up on PCA due to poor efficiency.

5.3 Mean Face

We obtained a direct visualization of the faces from each category using the training data Figure 9. A mean face is calculated by summing all the images.

5.4 Whitening

We also tried to apply whitening to our dataset. The whitening operation takes the data in the eigenbasis and divides every dimension by the eigenvalue to normalize the scale. The geometric interpretation of this transformation is that if the input data is a multivariable gaussian, then the whitened data will be a gaussian with zero mean and identity covariance matrix[10].

Similar to PCA, the whitening also needs to compute the covariance matrix which slows down the entire process.

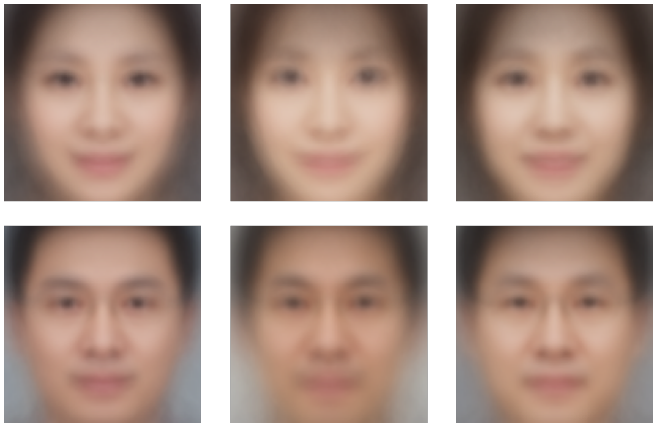

Figure 9: Mean faces. From left to right: Chinese, Japanese, Korean. From top to down: Female, male.

6 Conclusion

Machine learning methods are powerful in differentiating facial images of different countries, here we experimented several kinds of classifying methods and reported much higher prediction accuacy than baseline. The highest test accuracy we can get is 89.2% for 3 classes and 83.5% for 6 classes. The most effective pre-processing method is the combining of grayscale and mean subtraction after cropping faces.

Despite a high test accuracy on our dataset, CNN only predicts 61.3% accuracy on other new images, indicating it is overfitting due to the limited size and coverage of our data. In the future we can try getting more diverse data and running our model on GPU to see if we can find a better model.

References

[1] Fu, Siyao, Haibo He, and Zeng-Guang Hou. Learning race from face: a survey. IEEE transactions on pattern analysis and machine intelligence 36.12 (2014): 2483-2509.

[2] Tensorflow neural network library, `https://www.tensorflow.org/versions/r0.11/tutorials/mnist/tf/#tutorial-files`

[3] University of Tokyo, `http://www.u-tokyo.ac.jp/en/index.html`

[4] Waseda University, `https://www.waseda.jp/top/en`

[5] Seoul National University, `http://www.snu.ac.kr/index.html`

[6] Shanghai Jiaotong University, `http://www.snu.ac.kr/index.html`

[7] Sun Yat-Yen University, `http://www.sysu.edu.cn/2012/en/index.htm`

[8] Google Custom Search API, `https://developers.google.com/custom\discretionary{-}{}{}search/`

[9] Haas Cascades face detection library, `http://docs.opencv.org/trunk/d7/d8b/tutorial_py_face_detection.html`

[10] Stanford CS231N course notes, `http://cs231n.github.io/neural-networks-2/`

[11] MLA Phillips, P. Jonathon, et al. "An other-race effect for face recognition algorithms." ACM Transactions on Applied Perception (TAP) 8.2 (2011): 14.